Questions : – Système de Hilbert, c'est quoi ? | – Écrire les règles de la négation de la déduction naturelle ?
– Comparer la déduction naturelle ?... | – C'est quoi la logique intuitionniste ?

918 — Systèmes formels de preuve en logique du premier ordre. Exemples.

Un <u>système formel de preuve</u> est un ensemble de règles formelles portant sur des formules d'une certaine logique (ici, premier ordre) qui formalise les démonstrations dans le cadre de cette logique.
Parmi les premiers systèmes de preuve à apparaître furent les <u>systèmes de Hilbert</u> basés sur l'idée de système axiomatique : beaucoup d'axiomes, et quelques règles de déduction permettant de construire de nouvelles formules. Nous nous concentrerons ici sur des systèmes avec peu d'axiomes : le calcul des séquents et la résolution. Ces systèmes sont adaptés pour obtenir des résultats théoriques sur les preuves, et peuvent être utilisés dans le cadre de la preuve assistée ou automatique.
On fixe un langage du premier ordre $\mathcal{L}$.

I) Le calcul des séquents : [DNR] p186-187

1) Description du système :

Déf 1 : Un <u>séquent</u> est une expression de la forme $\Gamma \vdash \Delta$ où Γ et Δ sont des multi-ensembles de formules du premier ordre.

Le calcul des séquents (appelé LK) porte sur ces séquents, en interprétant $\Gamma \vdash \Delta$ intuitivement comme "la conjonction des formules de Γ a pour conséquence la disjonction de celles de Δ".

Déf 2 : Le <u>système LK</u> (A formule, $\Gamma, \Delta, \Gamma', \Delta'$ ensembles finis de formules)

* Axiomes : $\dfrac{}{\bot \vdash}\bot_g \qquad \dfrac{}{A \vdash A}\text{ax}$

* Règles structurelles : $\dfrac{\Gamma \vdash \Delta}{\Gamma, A \vdash \Delta}\text{aff}_g \qquad \dfrac{\Gamma \vdash \Delta}{\Gamma \vdash \Delta, A}\text{aff}_d$

$\dfrac{\Gamma, A, A \vdash \Delta}{\Gamma, A \vdash \Delta}\text{contr}_g \qquad \dfrac{\Gamma \vdash \Delta, A, A}{\Gamma \vdash \Delta, A}\text{contr}_d$

* Règles des connecteurs : $\dfrac{\Gamma, A, B \vdash \Delta}{\Gamma, A \wedge B \vdash \Delta}\wedge_g \qquad \dfrac{\Gamma \vdash A, \Delta \quad \Gamma \vdash B, \Delta}{\Gamma \vdash A \wedge B, \Delta}\wedge_d$

$\dfrac{\Gamma, A \vdash \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \vee B \vdash \Delta}\vee_g \qquad \dfrac{\Gamma \vdash A, B, \Delta}{\Gamma \vdash A \vee B, \Delta}\vee_d$

$\dfrac{\Gamma \vdash A, \Delta \quad \Gamma, B \vdash \Delta}{\Gamma, A \to B \vdash \Delta}\to_g \qquad \dfrac{\Gamma, A \vdash B, \Delta}{\Gamma \vdash A \to B, \Delta}\to_d$

$\dfrac{\Gamma \vdash A, \Delta}{\Gamma, \neg A \vdash \Delta}\neg_g \qquad \dfrac{\Gamma, A \vdash \Delta}{\Gamma \vdash \neg A, \Delta}\neg_d$

* Règles des quantificateurs :

$\dfrac{\Gamma, A[t/x] \vdash \Delta}{\Gamma, \forall x A \vdash \Delta}\forall_g \qquad \dfrac{\Gamma \vdash A, \Delta}{\Gamma \vdash \forall x A, \Delta}\forall_d \ (x \notin VL(\Gamma, \Delta))$ (variables libres)

$\dfrac{\Gamma, A \vdash \Delta}{\Gamma, \exists x A \vdash \Delta}\exists_g \ (x \notin VL(\Gamma, \Delta)) \qquad \dfrac{\Gamma \vdash A[t/x], \Delta}{\Gamma \vdash \exists x A, \Delta}\exists_d$

* Règle de coupure : $\dfrac{\Gamma \vdash A, \Delta \quad \Gamma', A \vdash \Delta'}{\Gamma, \Gamma' \vdash \Delta, \Delta'}\text{cut}$

Déf 3 : Un séquent $\Gamma \vdash \Delta$ est dit <u>prouvable</u> s'il existe un arbre étiqueté par des séquents tel que $\Gamma \vdash \Delta$ est à la racine et chaque arête correspond à l'application d'une règle de LK, et les feuilles correspondent à des axiomes.

Exples 4 : ① $\vdash \neg\neg A \to A$ est prouvable :

$$\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\text{ax}}{\vdash A, \neg A}\neg_d}{\neg\neg A \vdash A}\neg_g}{\vdash \neg\neg A \to A}\to_d$$

② $\vdash \neg \forall x A \to \exists x (\neg A)$ est prouvable :

$$\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{\dfrac{}{A \vdash A}\text{ax}}{\vdash A, \neg A}\neg_d}{\vdash A, \exists x(\neg A)}\exists_d}{\vdash \forall x A, \exists x(\neg A)}\forall_d \ (x \notin VL(\exists x (\neg A)))}{\neg \forall x A \vdash \exists x(\neg A)}\neg_g}{\vdash \neg \forall x A \to \exists x(\neg A)}\to_d$$

– Vous connaissez un système de preuve correspondant à la logique intuitionniste ?
– Cor 9. Vous pouvez en dire plus ? Résultat de ... (...)
–(2) pas clair. Expliquez ?
– Continuité en a : $\forall \varepsilon > 0, \exists \delta > 0, \forall x, |x - a| < \delta \Rightarrow |f(x) - f(a)| < \varepsilon$
Traduire sous CNF

– Montrer le pb est décidable [Entrée : formule de la forme $\exists \vec{x} \forall y F$ (F sans quant.) / Sortie : oui ssi admet un modèle]
(ss-pb de SAT (qui est indécidable au 1er ordre))

– Que fait l'algo d'unification ? Quelle est sa complexité ?
– Montrer que $\nexists \varphi$ / $\mathcal{M} \models \varphi$ ssi $|\text{dom}(\mathcal{M})| < \infty$ (compacité)
[Par l'absurde]

[DNR] §1.3 p23 → 33

(1) Rq 5 : La déduction naturelle est un autre système assez similaire et équivalent à LK, même si moins symétrique dans ses règles. Il est plus facile à manier pour des preuves "à la main", mais moins pour les résultats théoriques et l'automatisation.

Thm 6 : Correction et complétude de LK :
Pour tous ensembles finis de formules Γ et Δ, on a :
$(\Gamma \vdash \Delta$ est prouvable$)$ ssi $\left(\bigwedge_{F \in \Gamma} F \models \bigvee_{G \in \Delta} G \right)$

2) Élimination des coupures

[DNR] p200

Thm 7 : (Élimination des coupures)
Pour tout arbre de preuve de $\Gamma \vdash \Delta$ dans LK, on peut en construire un autre (algorithmiquement) qui n'utilise pas la règle de coupure.

Ce théorème a de nombreuses conséquences théoriques et pratiques :

[DNR] p215

Cor 8 : (Propriété de la sous-formule) Si $\Gamma \vdash \Delta$ est prouvable alors il en existe un arbre de preuve ne contenant que des formules de la forme $F[t_1/x_1, \dots, t_n/x_n]$ avec F une sous-formule d'une formule de $\Gamma \cup \Delta$.

Ce corollaire peut être utilisé pour la recherche de preuves automatiques.

Cor 9 : la logique propositionnelle est décidable.

(2) Rq 10 : La méthode des tableaux utilise l'élimination des coupures pour donner un algorithme montrant la semi-décidabilité de la logique du premier ordre.

[DNR] p216 DEV

Cor 11 : (Lemme d'interpolation de Craig)
Si $\vdash A \to B$ est prouvable, alors il existe une formule C dont les symboles non-logiques apparaissent à la fois dans A et B telle que $\vdash A \to C$ et $\vdash C \to B$ sont prouvables.

→ Prévoir quels cas seront démontrés.

II Résolution et Méthode de Herbrand

Ici pour prouver que φ est conséquence de Σ,
On pose d'abord $\tilde{E} = \Sigma \cup \{\neg \varphi\}$
qu'on transforme en un ens. de clauses équivalent E,
puis on applique soit la méthode de Herbrand,
soit les règles de la résolution, pour montrer
que E n'a pas de modèle (car $\Sigma \models \varphi$ ssi $\Sigma \cup \{\neg\varphi\}$ n'a pas de modèle)

$$\begin{array}{c} \Sigma, \varphi \\ \downarrow \\ \tilde{E} = \Sigma \cup \{\neg \varphi\} \\ \downarrow \\ E \equiv \tilde{E} \\ \swarrow \quad \searrow \\ \emptyset \qquad \emptyset \end{array}$$

NB : On s'appuiera ici sur la correction et la complétude du calcul des séquents.

1) Mise sous forme de clauses [L,R] p63-68.

On passe d'un ensemble d'énoncés à un ensemble de clauses par les opérations suivantes.

→ Mise sous forme prénexe
→ Mise sous CNF
→ Mise sous forme de Skolem
(ie remplacement des variables existentiellement quantifiées par des nouveaux symboles de fonctions)
→ Omettre les quantificateurs universels

ex $\varphi : \neg \exists x ((\neg Q x) \Rightarrow \forall y \, P y)$
↳ $\forall x \, \exists y \, \neg((\neg Q x) \Rightarrow P y)$
↳ $\forall x \, \exists y \, (\neg Q x) \wedge (\neg P y)$
↳ $\forall x \, (\neg Q x) \wedge (\neg P f(x))$
↳ $\{\neg Q x \, ; \, \neg P f(x)\}$

Déf Soit C une clause. Soient $L_1, \dots, L_p$ ses littéraux, $x_1 \dots x_n$ ses variables.
La signification de C est $\mathcal{S}(C) = \forall x_1 \forall x_2 \dots \forall x_n, L_1 \vee L_2 \dots \vee L_p$.

Soit E un ensemble de clauses qu'on note $C_1 \dots C_q$.
La signification de E est $\mathcal{S}(E) = \mathcal{S}(C_1) \wedge \mathcal{S}(C_2) \dots \wedge \mathcal{S}(C_q)$

Pté Soit E la mise sous forme de clauses d'un ensemble d'énoncés $\tilde{E}$.
Alors E admet un modèle ssi $\tilde{E}$ admet un modèle.

ex $\tilde{E} = \{\exists x (P x \to \forall y \, P y) \, ; \, \forall x (P x \vee Q x)\} \cup \{\neg (\exists x ((\neg Q x) \to \forall y \, P y))\}$
$E = \{\{\neg P c \, ; \, P y\}, \{P x \, ; \, Q x\}, \{\neg Q x \, ; \, \neg P f(x)\}\}$
et $\mathcal{S}(\{\neg P c \, ; \, P y\}) = \forall x \forall y, \neg P c \vee P y$.

(1) : Détailler

2) La méthode de Herbrand

Déf Si la clause $C = \{L_1, \dots, L_k\}$ fait intervenir les variables $x_1, \dots, x_n$, on définit l'ensemble des instances de C comme étant

$$\mathrm{Inst}(C) = \{(L_1 \vee \dots \vee L_k)[\sigma] \mid \sigma \text{ substitue à } x_1, \dots, x_n \text{ des termes clos}\}$$

ex Pour $C = \{\neg P_c, P_y\}$, $\mathrm{Inst}(C) = \{\neg P_c \vee P_t \mid t \text{ terme clos}\}$

Pté On considère $\hat{E}$ l'ensemble de clauses de la logique propositionnelle obtenu en remplaçant chaque formule atomique de $\mathrm{Inst}(E)$ par une variable propositionnelle lui correspondant.
E admet un modèle ssi $\hat{E}$ est satisfiable.

ex Si $\mathcal{L} = \{c, f\}$, $\mathrm{Inst}(C) = \bigcup_{n \in \mathbb{N}} \{\neg P_c \vee P_{f^n(c)}\}$

$$\hat{E} = \bigcup_{n\in\mathbb{N}} \neg V_{P_c} \vee V_{P_{f^n(c)}} \cup \bigcup_{n\in\mathbb{N}} \neg V_{Q_{f^n(c)}} \cup \bigcup_{(n,m)\in\mathbb{N}^2} V_{P_{f^n(c)}} \vee V_{Q_{f^m(c)}} \cup \bigcup_{n\in\mathbb{N}^*} \neg V_{P_{f(x)}}$$

$$\dfrac{\dfrac{\dfrac{\neg V_{P_c} \vee V_{P_{f(c)}} \qquad V_{P_c} \vee V_{Q_c}}{V_{P_{f(c)}} \vee V_{Q_c}} \qquad \neg V_{Q_c}}{V_{P_{f(c)}}} \qquad \neg V_{P_{f(c)}}}{\emptyset}$$

3) La méthode de résolution [DNR] p264 → 267.

Jà pour MQ E n'a pas de modèle on cherche à dériver E en $\emptyset$ selon les 2 règles suivantes.

Déf Soient C_1, C_2 des clauses ; L_1, L_2 des littéraux, σ une substitution.

- $\dfrac{C_1, L_1 \qquad C_2, L_2}{C_1[\sigma], C_2[\sigma]}$ rés si $L_1[\sigma] = \overline{L_2[\sigma]}$ "résolution"

- $\dfrac{C_1, L_1, L_2}{C_1[\sigma], L_1[\sigma]}$ cont si $L_1[\sigma] = L_2[\sigma]$ "contraction"

ex

$$\dfrac{\dfrac{\dfrac{\{\neg P_c, P_y\} \qquad \{P_x, Q_x\}}{\{P_y, Q_c\}}\text{rés} \qquad \{\neg Q_x\}}{\{P_y\}}\text{rés} \qquad \{\neg P f(x)\}}{\emptyset}\text{rés}$$

avec $\sigma = x \mapsto c$ $L_1 = \neg P_c$ $\overline{L_2} = \neg P_x$

avec $\sigma = x \mapsto c$, $L_1 = Q_c$ $\overline{L_2} = Q_x$.

avec $\sigma = y \mapsto f(x)$
$L_1 = P_y$ $\overline{L_2} = Pf(x)$

lemmes • Si $\Sigma \vdash A \vee B$ et $\Sigma \vdash \neg A$ en calcul des séquents alors $\Sigma \vdash B$ en calcul des séquents $\qquad \dfrac{\Sigma \vdash A \vee B \quad \Sigma \vdash \neg A}{\Sigma \vdash B}$ lemme

• Soit Σ un ens. d'énoncés. Soit A une formule de variables libres $x_1 - x_n$. Soit σ une substitution.
Si $\Sigma \vdash \forall x_1 - \forall x_n A$, alors $\Sigma \vdash A[\sigma]$. $\qquad \dfrac{\Sigma \vdash \forall x_1 - \forall x_n A}{\Sigma \vdash A[\sigma]}$ part.

Pté Soit E un ensemble de clauses.
Si E est inconsistant (ie E se dérive en $\emptyset$ en résolu)
alors E est contradictoire (ie n'a pas de modèle) — **Correction de la résolution**

lemme Soit E un ensemble de clauses et L un littéral clos.
Si $E \cup \{L\} \vdash \emptyset$ (en résolution) alors $E \vdash \emptyset$ ou $E \vdash L'$ où $L'[\sigma] = \overline{L}$ pour certain σ.

Pté Soit E un ensemble de clauses. — **Complétude de la résolution**
Si E est consistant (ie $E \nvdash \emptyset$)
alors E n'est pas contradictoire (ie admet 1 modèle)

DEV

4) Unification [DNR] p248. (à quelques modif. près)

Pour appliquer les règles rés et cont, il nous faut un unificateur de littéraux, de termes. L'unification traite ce problème. On donne ci-dessous un algorithme renvoyant l'unificateur le plus général de deux littéraux.

```
UNIF (L1, L2)
  E ← {L1 ~ L2} ; σ ← id
  Tant que E ≠ Ø
     Choisir e ∈ E
     Comparer e avec
       | x ~ x  → E ← E\{e}
       | x ~ u ou u ~ x → Si x apparaît dans u
                          alors erreur "Occur-check"
                          sinon E ← E\{e} [x := u]
                                σ ← [x := u] ∘ σ
       | f(t1,...tp) ~ g(s1,...sq) → Si f ≠ g
                          alors erreur "clash fonction"
                          sinon E ← E\{e} ∪ {ti ~ si | i ∈ [1,p]}
       | ¬A ~ B ou B ~ ¬A → erreur "clash ¬"
       | ¬A ~ ¬B → E ← E\{e} ∪ {A ~ B}
       | R(t1,...tp) ~ Q(s1,...sq) → Si R ≠ Q
                          alors erreur "clash relation"
                          sinon E ← E\{e} ∪ {si ~ ti | i ∈ [1,p]}
  retourner σ.
```

COMPLÉTUDE DE LA RÉSOLUTION — David Nour Raffalli p267

Idée Intuitivement la complétude d'un système de preuve assure que tout ce qui est vrai, vrai dans les modèles, admet une preuve.
En général il s'agit de MQ si tout modèle de Σ satisfait φ ($\Sigma \vDash \varphi$) alors on peut prouver que φ est csq de Σ ($\Sigma \vdash \varphi$).
Ici, en résolution, si $\Sigma \vDash \varphi$ on sait que $\Sigma \cup \{\neg\varphi\}$ n'a pas de modèle, et donc la mise sous forme de clause E non plus, plus précisém^t $\mathcal{S}(E)$ n'a pas de modèle. (où $\mathcal{S}$ désigne la significa°).
On dira seulement que E est contradictoire.
Il faut s'assurer que la résolution le montre, c-à-d que E se dérive en vide en résolution, c-à-d que E est inconsistant.

Pté Soit E un ensemble de clauses.
Si E est contradictoire, alors E est inconsistant. (complétude)

Preuve : On le montre par la contraposée, c-à-d qu'on suppose E consistant (ne se dérivant pas en $\emptyset$ en résolution), et on va MQ E n'est pas contradictoire en exhibant un modèle, dit de Herbrand.

On suppose que le langage sur lequel sont écrites les formules contient au moins un symbole de constante, afin d'assurer que l'ensemble des termes clos soit non vide.

Puisque E est consistant on peut considérer l'ensemble Z des ensembles de clauses contenant E et consistants, qui est alors non vide.

$$Z = \{ F \text{ ens. de clauses} \mid E \subset F,\ F \text{ consistant} \} \neq \emptyset \text{ car } E \in Z.$$

Montrons que Z est un ensemble inductif pour l'inclusion, c-à-d que chaque chaîne admet un majorant.
Pour une chaîne S de Z on considère U l'union de tous ses élm^t. Nec $E \subset U$. Reste à vérifier que U est consistant. Par l'absurde si U est inconsistant on peut dériver la clause vide à partir des clauses de U, à partir d'1 nombre fini de clauses de U m̂. Or cela implique qu'elles sont toutes dans un m̂ élm^t $S_0 \in S$.
[⚠ vrai ici car S est une chaîne et U une union croissante]

Mais cela est absurde car $S_0 \in S \subset Z$ est consistant.
On en déduit que U est bien consistant, et donc $U \in Z$ et Z est inductif.

Cela nous permet d'utiliser le lemme de Zorn, qui assure l'existence de E_0 un élément maximal de Z.

Utilisons cette maximalité : considérons A une formule atomique close.
Si $A \notin E_0$ et $\neg A \notin E_0$, la maximalité de E_0 assure que $E_0 \cup \{A\}$ et $E_0 \cup \{\neg A\}$ sont inconsistants, c-à-d on peut dériver la clause vide à partir de $E_0 \cup \{\neg A\}$ et de $E_0 \cup \{A\}$.
Par le lemme technique qui suit cela implique l'existence de A' et A'' filtrant resp. A et $\neg A$, qu'on peut dériver à partir de E_0. A' et $\overline{A''}$ étant unifiables (en A), on pourrait dériver $\emptyset$ à partir de E_0 : ABSURDE !
On est donc assuré que chaque formule close A vérifie $A \in E_0$ ou $\neg A \in E_0$.

On construit maintenant le modèle de Herbrand $\mathcal{M}$

→ de domaine M = l'ensemble des termes clos.
→ pour c symbole de cst. $c^{\mathcal{M}}$ est $c \in M$
→ pour f —— f' $f^{\mathcal{M}}$ est $\begin{pmatrix} M^k \longrightarrow M \\ t_1 \ldots t_k \longmapsto f(t_1, \ldots, t_k) \end{pmatrix}$ le terme qui s'écrit $f(t)$ (noter $\bar{t}$)
→ pour R —— relation k-aire on pose

$$R^{\mathcal{M}} = \{ (t_1, \ldots, t_k) \in M^k \mid \underbrace{R(t_1, \ldots, t_k)}_{\text{formule atomique close}} \in E_0 \}$$

Montrons que $\mathcal{M}$ est modèle de E_0 (de $\mathcal{S}(E_0)$) et donc de E ($\mathcal{S}(E)$).
$\mathcal{S}(E_0)$ s'écrit comme une conjonction de clauses.
Soit C l'une de ses clauses, on l'écrit $\{A_i \mid i \in [1..p]\} \cup \{\neg B_j \mid j \in [1..q]\}$.
Alors $\mathcal{S}(C)$: $\forall x_1 \ldots \forall x_n\ A_1 \vee A_2 \ldots \vee A_p \vee \neg B_1 \vee \ldots \vee \neg B_q$.
Si $\mathcal{M} \nvDash \mathcal{S}(C)$, il existe $(t_i)_{i \in [1..n]} \in M^n$, tel qu'avec la signature $\sigma = x_i \mapsto t_i$, on ait $\mathcal{M}[\sigma] \nvDash A_1 \vee \ldots \vee A_p \vee \neg B_1 \ldots \vee \neg B_q$.
En particulier pour $i \in [1..p]$ $\mathcal{M}[\sigma] \nvDash A_i$
$j \in [1..q]$ $\mathcal{M}[\sigma] \nvDash \neg B_j$ soit $\mathcal{M}[\sigma] \vDash B_j$.

Par construction cela signifie que $\forall i \in [1..p]$ $A_i[\sigma] \notin E_0$
$\forall j \in [1..q]$ $B_j[\sigma] \in E_0$.

En effet A_i étant atomique on peut l'écrire $R(x_{i_1}, \ldots x_{i_k})$.
Alors par définition : $\mathcal{M}[\sigma] \vDash A_i$ ssi $(x_{i_1}, \ldots, x_{i_k})[\sigma] \in R^{\mathcal{M}}$
ssi $(t_{i_1}, \ldots, t_{i_k}) \in R^{\mathcal{M}}$
ssi $\underbrace{R(t_{i_1}, \ldots t_{i_k})}_{= A_i[\sigma]} \in E_0$

Mais alors E_0 est inconsistant, en effet on a la preuve :

$$\dfrac{\dfrac{\dfrac{A_1, A_2, \dots, A_p, \neg B_1, \neg B_2 \dots \neg B_q \qquad \neg A_1[\sigma]}{A_2[\sigma] \dots A_p[\sigma]\ \neg B_1[\sigma] \dots \neg B_q[\sigma]}\,\mathrm{res}(\sigma)}{\neg B_1[\sigma] \dots \neg B_q[\sigma]}\,\mathrm{res}(id)^{p-1}}{\emptyset}\,\mathrm{res}(id)^q$$

$\mathrm{res}(id)^{p-1}$ en utilisant successivement $\neg A_2[\sigma], \neg A_3[\sigma] \dots \neg A_p[\sigma]$

$\mathrm{res}(id)^q$ en utilisant succ. $B_1[\sigma], B_2[\sigma] \dots B_q[\sigma]$.

C'est absurde ! Donc $\mathcal{M} \models \mathcal{S}(C)$.

Ceci étant pour toute clause C de E_0, $\mathcal{M} \models \mathcal{S}(E_0)$.

Enfin comme $E \subset E_0$ $\mathcal{M} \models \mathcal{S}(E)$, donc E admet bien un modèle, et n'est pas contradictoire.

Lemme

Soit E un ensemble de clauses. Soit L un littéral clos.
Si $\emptyset$ se dérive à partir de $E \cup \{L\}$ (i.e. $E \cup \{L\} \vdash_{res} \emptyset$)
alors il existe un littéral qui filtre $\bar{L}$ tq $E \vdash_{res} L$ ou $E \vdash_{res} \emptyset$.

Et plus de fiches sur :
perso.eleves.ens-rennes.fr/~afalq494/

:)

[DNR] David, Nour, Raffalli
Introduction à la logique
2ème éd (Dunod)

[L/R] Lassaigne/Rougemont
Logique et fondem^t de l'info. (Hermès)

LEMME D'INTERPOLATION DE CRAIG

[DNR]
Sect° 5.5.4
Thm 5.5.21

Si A est une formule, on note $V(A) = \{$symb de $\mathcal{L}$ dans $A\} \cup \{$var libres de $A\}$ (le <u>vocabulaire</u> de A). Quitte à remplacer les symboles de fonction et de constante par des relations, on peut supposer que $\mathcal{L}$ ne contient que des symboles de relation. (cf exercice 2.18 de [DNR], par exemple)

<u>Thm</u> : Si $\mathcal{L}$ ne contient que des symboles de relation, et $A \vdash_{LK} B$, alors il existe C tel que $V(C) \subseteq V(A) \cap V(B)$, et $A \vdash_{LK} C$ et $C \vdash_{LK} B$.

<u>Dém</u> : On va démontrer par induction sur les arbres de preuve <u>sans coupure</u> la propriété suivante :

$$\mathcal{P}\left(\dfrac{\vdots}{\Gamma \vdash \Delta}\right) = \text{" Pour tous } \Gamma_1 \sqcup \Gamma_2 = \Gamma,\ \Delta_1 \sqcup \Delta_2 = \Delta, \text{ il existe une formule } F \text{ satisfaisant :}$$

$$H(\Gamma_1, \Delta_1, \Gamma_2, \Delta_2) = \begin{cases} (a)\ V(F) \subseteq V(\Gamma_1, \Delta_1) \cap V(\Gamma_2, \Delta_2) \\ (b)\ \Gamma_1 \vdash_{LK} F, \Delta_1 \\ (c)\ \Gamma_2, F \vdash_{LK} \Delta_2 \end{cases} \text{"}$$

(<u>Remarque</u> : Il suffira alors d'appliquer $\mathcal{P}\left(\dfrac{\vdots}{A \vdash B}\right)$ avec
$\Gamma_1 = A$, $\Gamma_2 = \emptyset$, $\Delta_1 = \emptyset$, $\Delta_2 = A$)

On ne fait la démonstration que pour certains cas significatifs :

Pas obligé de faire les deux

* $\boxed{\mathcal{P}\left(\dfrac{}{\bot \vdash}\ \bot_g\right)}$: On a toujours $\Delta_1 = \Delta_2 = \emptyset$.
  - → <u>Si $\Gamma_1 = \{\bot\}$, $\Gamma_2 = \emptyset$</u> : On pose $F = \bot$. Alors (a), (b) et (c) sont clairement satisfaits.
  - → <u>Si $\Gamma_1 = \emptyset$, $\Gamma_2 = \{\bot\}$</u> : On pose $F = \bot \to \bot$, idem.
* $\boxed{\mathcal{P}\left(\dfrac{}{A \vdash A}\ ax\right)}$:
  - → <u>Si $\underbrace{(\Gamma_1, \Delta_1, \Gamma_2, \Delta_2)}_{:= \Sigma} = (A, A, \emptyset, \emptyset)$</u> : $F = \bot \to \bot$
  - → <u>Si $\Sigma = (A, \emptyset, \emptyset, A)$</u> : $F = A$
  - → <u>Si $\Sigma = (\emptyset, A, A, \emptyset)$</u> : $F = \neg A$
  - → <u>Si $\Sigma = (\emptyset, \emptyset, A, A)$</u> : $F = \bot \to \bot$

* $\boxed{\mathcal{P}\left(\dfrac{\dfrac{\vdots}{\Gamma\vdash A,\Delta} \qquad \dfrac{\vdots}{\Gamma,B\vdash\Delta}}{\Gamma,A\to B\vdash\Delta}\to_g\right)}$: Il y a deux cas : "$A\to B$" $\in\Gamma_1$ ou "$A\to B$" $\in\Gamma_2$. On n'en présente qu'un des deux :

→ Supp. que $A\to B\in\Gamma_1$: Posons $\tilde{\Gamma}_1 = \Gamma_1\setminus$"$A\to B$".

D'après $\mathcal{P}\left(\dfrac{\vdots}{\Gamma\vdash A,\Delta}\right)$, il existe F_g qui satisfait $H(\tilde{\Gamma}_1,(\Delta_1,A),\Gamma_2,\Delta_2)$.

$\mathcal{P}\left(\dfrac{\vdots}{\Gamma,B\vdash\Delta}\right)$, il existe F_d qui satisfait $H((\tilde{\Gamma}_1,B),\Delta_1,\Gamma_2,\Delta_2)$

Posons $F=F_g\vee F_d$. Alors $V(F)=V(F_g)\cup V(F_d)$.

F satisfait ⓐ
- On a $V(F_g)\subseteq V(\Gamma_1,\Delta_1)$ et $V(F_d)\subseteq V(\Gamma_2,\Delta_2)$, donc $V(F)\subseteq V(\Gamma_2,\Delta_2)$.
- D'autre part, $V(F_g)\cup V(F_d)\subseteq V(\tilde{\Gamma}_1,\Delta_1,A,B)=V(\tilde{\Gamma}_1,\Delta_1,A\to B)=V(\Gamma_1,\Delta_1)$.
- Donc $V(F)\subseteq V(\Gamma_1,\Delta_1)\cap V(\Gamma_2,\Delta_2)$ ⇝ $\boxed{\text{ⓐ est satisfait pour } F}$

F satisfait ⓑ

On donne un arbre de preuve pour $\Gamma_1\vdash F,\Delta_1$:

$$\dfrac{\dfrac{\dfrac{\dfrac{ⓑ\text{ pour }F_g}{\tilde{\Gamma}_1\vdash A,F_g,\Delta_1}}{\tilde{\Gamma}_1\vdash A,F_g,F_d,\Delta_1}\text{aff}_d \qquad \dfrac{\dfrac{ⓑ\text{ pour }F_d}{\tilde{\Gamma}_1,B\vdash F_d,\Delta_1}}{\tilde{\Gamma}_1,B\vdash F_g,F_d,\Delta_1}\text{aff}_d}{\tilde{\Gamma}_1,A\to B\vdash F_g,F_d,\Delta_1}\to_g}{\Gamma_1\vdash F,\Delta_1}\vee_d$$

F satisfait ⓒ

On donne un arbre de preuve pour $\Gamma_2,F\vdash\Delta_2$:

$$\dfrac{\dfrac{ⓒ\text{ pour }F_g}{\Gamma_2,F_g\vdash\Delta_2} \qquad \dfrac{ⓒ\text{ pour }F_d}{\Gamma_2,F_d\vdash\Delta_2}}{\Gamma_2,F\vdash\Delta_2}\vee_g$$

→ L'autre cas ($A\to B\in\Gamma_2$) est similaire en posant $F=F_g\wedge F_d$.

* $\boxed{\mathcal{P}\left(\dfrac{\dfrac{\vdots}{\Gamma,A[t/x]\vdash\Delta}}{\Gamma,\forall xA\vdash\Delta}\forall_g\right)}$: Encore une fois, deux cas, on n'en fait qu'un. Comme $\mathcal{L}$ ne contient que des symboles de relations, le terme t est une variable y.

→ Supp que $\forall xA\in\Gamma_1$: Posons $\tilde{\Gamma}_1=\Gamma_1\setminus\{\forall xA\}$

D'après $\mathcal{P}\left(\dfrac{\vdots}{\Gamma,A[y/x]\vdash\Delta}\right)$, il existe G satisfaisant $H((\tilde{\Gamma}_1,A[y/x]),\Delta_1,\Gamma_2,\Delta_2)$.

Le problème qui peut se produire ici est que y peut être une variable libre dans G, mais pas dans $\Gamma,\forall xA,\Delta$, et donc ne pas être autorisée comme variable libre dans F. On distingue trois cas selon le statut de y :

- Si $y\notin V(\Gamma_1,\Delta_1)$: Posons $F=\forall y\, G$. Alors ⓐ est satisfait pour F, et :

$$\dfrac{\dfrac{\dfrac{ⓑ\text{ pour }G}{\tilde{\Gamma}_1,A[y/x]\vdash G,\Delta_1}}{\Gamma_1\vdash G,\Delta_1}\forall_g}{\Gamma_1\vdash F,\Delta_1}\forall_d\ (\text{car } y\notin VL(\Gamma_1,\Delta_1)) \qquad \dfrac{\dfrac{ⓒ\text{ pour }G}{\Gamma_2,G\vdash\Delta_2}}{\Gamma_2,F\vdash\Delta_2}\forall_g$$

- Si $y\notin V(\Gamma_2,\Delta_2)$: Posons $F=\exists y\, G$ ⇝ De même que ci-dessus.
- Si $y\in V(\Gamma_1,\Delta_1)\cap V(\Gamma_2,\Delta_2)$: On pose $F=G$ (cette fois, c'est licite de garder y en var libre), le reste suit immédiatement.

∎

Remarques :

- La restriction sur $\mathcal{L}$ n'est pas nécessaire, mais ça a l'air galère de le faire sans ça, donc je pense qu'il vaut mieux le présenter comme ça.
- Peut-être s'attendre à devoir développer un cas non fait ici ? C'est pas trop compliqué de deviner F une fois qu'on l'a fait une fois ou deux.
- C'est constructif ! (À condition d'avoir un arbre de preuve, mais apparemment c'est utilisé en preuve automatique).